```
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN  VVV         VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN  VVV         VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN  VVV         VVV
CCC              OOO         OOO  NNNNNN      NNN  VVV         VVV
CCC              OOO         OOO  NNNNNN      NNN  VVV         VVV
CCC              OOO         OOO  NNNNNN      NNN  VVV         VVV
CCC              OOO         OOO  NNN   NNN   NNN  VVV         VVV
CCC              OOO         OOO  NNN   NNN   NNN  VVV         VVV
CCC              OOO         OOO  NNN   NNN   NNN  VVV         VVV
CCC              OOO         OOO  NNN      NNNNNN  VVV         VVV
CCC              OOO         OOO  NNN      NNNNNN  VVV         VVV
CCC              OOO         OOO  NNN      NNNNNN  VVV         VVV
CCC              OOO         OOO  NNN         NNN     VVV   VVV
CCC              OOO         OOO  NNN         NNN     VVV   VVV
CCC              OOO         OOO  NNN         NNN     VVV   VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN        VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN        VVV
   CCCCCCCCCCCC     OOOOOOOOO     NNN         NNN        VVV
```

```
**FILE**ID**RECLCTRL                              G 9




RRRRRRRR    EEEEEEEEEE    CCCCCCCC  LL              CCCCCCCC  TTTTTTTTTT  RRRRRRRR    LL
RRRRRRRR    EEEEEEEEEE    CCCCCCCC  LL              CCCCCCCC  TTTTTTTTTT  RRRRRRRR    LL
RR      RR  EE          CC          LL          CC                TT      RR      RR  LL
RR      RR  EE          CC          LL          CC                TT      RR      RR  LL
RR      RR  EE          CC          LL          CC                TT      RR      RR  LL
RR      RR  EE          CC          LL          CC                TT      RR      RR  LL
RRRRRRRR    EEEEEEEE    CC          LL          CC                TT      RRRRRRRR    LL
RRRRRRRR    EEEEEEEE    CC          LL          CC                TT      RRRRRRRR    LL
RR  RR      EE          CC          LL          CC                TT      RR  RR      LL
RR  RR      EE          CC          LL          CC                TT      RR  RR      LL
RR    RR    EE          CC          LL          CC                TT      RR    RR    LL              ....
RR    RR    EE          CC          LL          CC                TT      RR    RR    LL              ....
RR      RR  EEEEEEEEEE    CCCCCCCC  LLLLLLLLLL    CCCCCCCC        TT      RR      RR  LLLLLLLLLL      ....
RR      RR  EEEEEEEEEE    CCCCCCCC  LLLLLLLLLL    CCCCCCCC        TT      RR      RR  LLLLLLLLLL      ....


LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL               II     SS
LL               II     SS
LL               II     SS
LL               II     SS
LL               II       SSSSSS
LL               II       SSSSSS
LL               II             SS
LL               II             SS
LL               II             SS
LL               II             SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
0 %TITLE  'VAX-11 CONVERT/RECLAIM'
0 MODULE  RECL$CTRL          ( IDENT='V04-000',
0                              OPTLEVEL=3
0                              ) =
0
1 BEGIN
1
1 !****************************************************************************
1 !*                                                                          *
1 !*   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                *
1 !*   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                 *
1 !*   ALL RIGHTS RESERVED.                                                   *
1 !*                                                                          *
1 !*   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED  *
1 !*   ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
1 !*   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
1 !*   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
1 !*   OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
1 !*   TRANSFERRED.                                                           *
1 !*                                                                          *
1 !*   THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
1 !*   AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
1 !*   CORPORATION.                                                           *
1 !*                                                                          *
1 !*   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
1 !*   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                *
1 !*                                                                          *
1 !*                                                                          *
1 !****************************************************************************
```

```
!++
!
! Facility:     VAX-11 CONVERT/RECLAIM
!
! Environment:
!               VAX/VMS Operating System
!
! Abstract:
!
!
! Contents:     SCAN_DATA_LEVEL
!               UPDATE_INDEX
!               REMOVE_BUCKET
!               ZERO_BUCKET
!               SWAP_BUFFERS
!
!--
!
!
! Author:       Keith B Thompson
!               Peter Lieberwirth       Creation date:  September-1981
!
!
! Modified by:
!
!       V03-007 JWT0176         Jim Teague              13-Apr-1984
!               Fix linkages to CONV$$WRITE_AREA_DESC and
!               CONV$$WRITE_KEY_DESC.
!
!       V03-006 KBT0395         Keith B. Thompson       29-Oct-1982
!               Add support for prologue 3 sidrs
!
!       V03-005 KBT0358         Keith B. Thompson       6-Oct-1982
!               Use new merged ctx definitions
!
!       V03-004 KBT0353         Keith B. Thompson       5-Oct-1982
!               Use new linkage definitions
!
!       V03-003 KBT0048         Keith Thompson          21-Apr-1982
!               Do not reclaim the last index record in a bucket
!
!       V03-002 KBT0041         Keith Thompson          3-Apr-1982
!               Add logic to swing index pointers if needed and fix index
!               save bucket logic
!
!       V03-001 KBT0010         Keith Thompson          16-Mar-1982
!               Fix a problem with end condition in update_index and
!               add a few lines of comments.
!
!****
```

```
1
1 PSECT
1          OWN      = _CONV$RECL_D  (PIC),
1          GLOBAL   = _CONV$RECL_D  (PIC),
1          PLIT     = _CONV$PLIT    (SHARE,PIC),
1          CODE     = _CONV$RECL_S  (SHARE,PIC);
1
1 LIBRARY 'SYS$LIBRARY:LIB.L32';
1 LIBRARY 'SRC$:CONVERT';
1
1 EXTERNAL ROUTINE
1          RECL$$GET_NEXT_BUCKET    : RL$JSB_REG_9  NOVALUE,
1          RECL$$BUCKET_EMPTY       : RL$JSB_REG_9,
1          RECL$$GET_DOWN_POINTER   : RL$JSB_REG_8,
1          RECL$$CHECK_LAST         : RL$JSB_REG_8,
1          RECL$$COMPARE_POINTER    : RL$JSB_REG_8,
1          RECL$$SWING_POINTER      : RL$JSB_REG_8  NOVALUE,
1          RECL$$REMOVE_INDEX_RECORD: RL$JSB_REG_8,
1          RECL$$WRITE_BUCKET       : RL$JSB_REG_9  NOVALUE,
1          CONV$$WRITE_AREA_DESC    : CL$WRITE_AREA_DESC    NOVALUE,
1          CONV$$WRITE_KEY_DESC     : CL$WRITE_KEY_DESC     NOVALUE;
1
1 FORWARD ROUTINE
1          UPDATE_INDEX             : RL$JSB_REG_9,
1          REMOVE_BUCKET            : RL$JSB_REG_9  NOVALUE,
1          ZERO_BUCKET              : RL$JSB_REG_9  NOVALUE,
1          RECL$$SWAP_BUFFERS       : RL$JSB_REG_9  NOVALUE;
1
1 EXTERNAL
1          CONV$AR_AREA_BLOCK;
1
```

```
1 %SBTTL  'SCAN_DATA_LEVEL'
1 GLOBAL ROUTINE  RECL$$SCAN_DATA_LEVEL : RL$JSB_REG_9 =
1 !++
1 !
1 ! Functional Description:
1 !
1 !       This routine sequentially read along the data level buckets
1 !       looking for an empty one.  If it finds one it trys to remove
1 !       the index to it then trys to remove it.
1 !
1 ! Calling Sequence:
1 !
1 !       RECL$$SCAN_DATA_LEVEL()
1 !
1 ! Input Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !
1 !       BUCKET
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !       none
1 !
1 ! Routine Value:
1 !
1 !       normal
1 !
1 ! Routines Called:
1 !
1 !       BUCKET_EMPTY
1 !       UPDATE_INDEX
1 !       REMOVE_BUCKET
1 !       SWAP_BUFFERS
1 !       GET_NEXT_BUCKET
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     DEFINE_CTX;
2     DEFINE_BUCKET;
2     DEFINE_KEY_DESC;
2
2     ! Loop untill the last bucket in chain if found.
2     ! If this bucket is the last in the chain don't do it (it is to
2     ! complaicated to reclaim this one bucket) instead go to the
2     !
3     WHILE ( NOT .BUCKET [ BKT$V_LASTBKT ] )
2     DO
3         BEGIN
```

```
3
3           ! If the bucket is empty the try to remove all traces of it
3           !
3           IF RECL$$BUCKET_EMPTY()
3           THEN
4               BEGIN
4
4               ! Remove the index record for this bucket
4               !
4               IF UPDATE_INDEX( .CTX [ CTX$L_CURRENT_VBN ] )
4               THEN
4
4                   ! If the update was successful remove the bucket itself
4                   !
4                   REMOVE_BUCKET()
4
4               ELSE
4
4                   ! If index could not be update then swap the buffers in order
4                   ! to save the previous bucket
4                   !
4                   RECL$$SWAP_BUFFERS()
4
4               END
3           ELSE
3
3               ! If the bucket is not empty then swap the buffers in order to save
3               ! the previous bucket
3               !
3               RECL$$SWAP_BUFFERS();
3
3           ! Get the next bucket
3           !
3           RECL$$GET_NEXT_BUCKET()
3
2           END;
2
2       RETURN RECL$_SUCCESS
2
1       END;
```

```
                                                .TITLE  RECL$CTRL VAX-11 CONVERT/RECLAIM
                                                .IDENT  \V04-000\

                                                .EXTRN  RECL$$GET_NEXT_BUCKET
                                                .EXTRN  RECL$$BUCKET_EMPTY
                                                .EXTRN  RECL$$GET_DOWN_POINTER
                                                .EXTRN  RECL$$CHECK_LAST
                                                .EXTRN  RECL$$COMPARE_POINTER
                                                .EXTRN  RECL$$SWING_POINTER
                                                .EXTRN  RECL$$REMOVE_INDEX_RECORD
                                                .EXTRN  RECL$$WRITE_BUCKET
                                                .EXTRN  CONV$$WRITE_AREA_DESC
                                                .EXTRN  CONV$$WRITE_KEY_DESC
                                                .EXTRN  CONV$AR_AREA_BLOCK
```

```
                                                       .PSECT  _CONV$RECL_S,NOWRT,  SHR,  PIC,2

                  1F      0D    A9  E8 00000 RECL$$SCAN_DATA_LEVEL::
                                                       BLBS    13(BUCKET), 3$                                ; 0165
                             0000G  30 00004           BSBW    RECL$$BUCKET_EMPTY                            ; 0171
                  11            50  E9 00007           BLBC    R0, 1$                                        ;
                          08    AA  DD 0000A           PUSHL   8(CTX)                                        ; 0177
                             0000V  30 0000D           BSBW    UPDATE_INDEX                                  ;
                  5E            04  C0 00010           ADDL2   #4, SP                                        ;
                  05            50  E9 00013           BLBC    R0, 1$                                        ;
                             0000V  30 00016           BSBW    REMOVE_BUCKET                                 ; 0182
                                03  11 00019           BRB     2$                                            ;
                             0000V  30 0001B 1$:       BSBW    RECL$$SWAP_BUFFERS                            ; 0197
                             0000G  30 0001E 2$:       BSBW    RECL$$GET_NEXT_BUCKET                         ; 0201
                                DD  11 00021           BRB     RECL$$SCAN_DATA_LEVEL                         ;
                  50            01  D0 00023 3$:       MOVL    #1, R0                                        ; 0205
                                    05 00026           RSB                                                   ; 0207

; Routine Size:  39 bytes,    Routine Base:  _CONV$RECL_S + 0000
;   212        0208  1  %SBTTL  'UPDATE_INDEX'
;   213        0209  1  ROUTINE UPDATE_INDEX ( VBN ) : RL$JSB_REG_9 =
;   214        0210  1  !++
;   215        0211  1  !
;   216        0212  1  ! Functional Description:
;   217        0213  1  !
;   218        0214  1  !       This routine updates the level above when a bucket on the lower level
;   219        0215  1  !       is deleted.  When called recursively, it updates the entire index.
;   220        0216  1  !
;   221        0217  1  ! Calling Sequence:
;   222        0218  1  !
;   223        0219  1  !       UPDATE_INDEX( VBN );
;   224        0220  1  !
;   225        0221  1  ! Input Parameters:
;   226        0222  1  !
;   227        0223  1  !       VBN     -       the VBN of the bucket being deleted on the lower level
;   228        0224  1  !
;   229        0225  1  ! Implicit Inputs:
;   230        0226  1  !
;   231        0227  1  !       BUCKET
;   232        0228  1  !       KEY_DESC
;   233        0229  1  !
;   234        0230  1  ! Output Parameters:
;   235        0231  1  !
;   236        0232  1  !       None.
;   237        0233  1  !
;   238        0234  1  ! Implicit Outputs:
;   239        0235  1  !
;   240        0236  1  !       None.
;   241        0237  1  !
;   242        0238  1  ! Routine Value:
;   243        0239  1  !
;   244        0240  1  !       SUCCESS or FAILURE
;   245        0241  1  !
;   246        0242  1  ! Routines Called:
;   247        0243  1  !
;   248        0244  1  !       GET_DOWN_POINTER
;   249        0245  1  !       COMPARE_POINTER
;   250        0246  1  !       SWING_POINTER
;   251        0247  1  !       REMOVE_INDEX_RECORD
;   252        0248  1  !       BUCKET_EMPTY
;   253        0249  1  !       UPDATE_INDEX
;   254        0250  1  !       REMOVE_BUCKET
;   255        0251  1  !       WRITE_BUCKET
;   256        0252  1  !       GET_NEXT_BUCKET
;   257        0253  1  !       SWAP_BUFFERS
;   258        0254  1  !
;   259        0255  1  ! Side Effects:
;   260        0256  1  !
;   261        0257  1  !       None.
;   262        0258  1  !
;   263        0259  1  !--
;   264        0260  1
;   265        0261  2      BEGIN
;   266        0262  2
;   267        0263  2      DEFINE_CTX;
;   268        0264  2      DEFINE_BUCKET;
```

```
DEFINE_KEY_DESC;
DEFINE_KEY_POINTER_GLOBAL;

LOCAL
    STATUS,
    NEXT_DATA_BUCKET;

! Assume success
!
STATUS = RECL$_SUCCESS;

! Return success if at level with root bucket
!
IF .BUCKET [ BKT$V_ROOTBKT ]
THEN
    RETURN .STATUS;

! Before we move up a level get the vbn of the next bucket (when this is
! the data level it will be important)
!
NEXT_DATA_BUCKET = .BUCKET [ BKT$L_NXTBKT ];

! Point the context at the next higher level in the tree
!
CTX = .CTX + CTX$K_BLN;

! Point to the new bucket
!
BUCKET = .CTX [ CTX$L_CURRENT_BUFFER ];

! Save the position in the index so we can come back
!
CTX [ CTX$L_SAVE_VBN ] = .CTX [ CTX$L_PREVIOUS_VBN ];

! Search all the buckets on the current level for a down pointer
!
DO
    BEGIN

    ! Is down pointer in current bucket?
    !
    IF RECL$$GET_DOWN_POINTER( .VBN )
    THEN
        BEGIN

        !++
        !
        ! Yes, we found the down pointer in the current bucket.
        ! Check to see if it is the last pointer in a bucket if so we
        ! can't reclaim it.
        ! If this is level 1 check to see if the next index pointer points
        ! to the next data bucket.  If it doesent we swing the current
        ! pointer to point to the next data bucket.  Otherwise we squish
        ! out the down pointer.  Then to see if squishing out the down
        ! pointer made the bucket reclaimable.  If it did, reclaim it after
        ! updating the index levels above this one.  If its not reclaimable
        ! just re-compress the index record following the deleted down
```

```
                    ! pointer and write the bucket back.
                    !
                    !--

                    ! If this is the last index record in the bucket then don't reclaim it
                    !
                    IF RECL$$CHECK_LAST()
                    THEN
                        BEGIN
                        STATUS = RECL$_FAILURE;
                        EXITLOOP
                        END;

                    IF .CTX [ CTX$B_LEVEL ] EQLU 1
                    THEN

                        ! Check to see if the next index pointer points to the
                        ! next data bucket
                        !
                        IF RECL$$COMPARE_POINTER( .NEXT_DATA_BUCKET )
                        THEN

                            ! If it does, simply remove the current index record
                            !
                            RECL$$REMOVE_INDEX_RECORD()

                        ELSE

                            ! If it doesent, swing the current index record to point
                            ! to the next data bucket
                            !
                            RECL$$SWING_POINTER( .NEXT_DATA_BUCKET )

                    ELSE

                        ! Squish out the index record in the current buffer
                        !
                        RECL$$REMOVE_INDEX_RECORD();

                    ! if this index bucket is empty then lets try to reclaim it!
                    !
                    IF RECL$$BUCKET_EMPTY()
                    THEN
                        BEGIN

                        ! If the index bucket is empty, try to update all the
                        ! index levels above.
                        ! If sucessful remove it.
                        !
                        IF STATUS = UPDATE_INDEX ( .CTX [ CTX$L_CURRENT_VBN ] )
                        THEN
                            BEGIN

                            ! If the update was successful remove the bucket
                            !
                            REMOVE_BUCKET();

```

```
                                ! Get the next bucket so we don't look at this one again
                                !
                                RECL$$GET_NEXT_BUCKET()

                                END
                            ELSE
                                BEGIN

                                ! If the update failed then we must reread the buffer since
                                ! it was modified
                                !
                                CTX [ CTX$L_NEXT_VBN ] = .CTX [ CTX$L_SAVE_VBN ];

                                ! Zero the current buffer vbn to force the read
                                !
                                CTX [ CTX$L_CURRENT_VBN ] = 0;

                                ! Get the saved previous bucket
                                !
                                RECL$$GET_NEXT_BUCKET();

                                EXITLOOP

                                END

                            END
                        ELSE
                            BEGIN

                            ! bucket is not empty so just write the current
                            ! buffer back, and return
                            !
                            RECL$$WRITE_BUCKET( CTX [ CTX$L_CURRENT_BUFFER ] );

                            EXITLOOP

                            END

                        END

                    ELSE

                        ! Down pointer is not in current buffer so read in the next bucket
                        ! in the horizontal chain.
                        !
                        ! However, if this is already the last bucket in this level, we
                        ! didn't find the down pointer, so return saying success, since
                        ! if there's no down pointer we can certainly reclaim the bucket
                        ! on the level below.
                        !
                        IF .BUCKET [ BKT$V_LASTBKT ]
                        THEN
                            BEGIN

                            ! If this bucket is the same as the save bucket then
                            ! don't bother to reread it
                            !
```

```
4                    IF .CTX [ CTX$L_CURRENT_VBN ] NEQU .CTX [ CTX$L_SAVE_VBN ]
4                    THEN
5                        BEGIN
5
5                        ! Before we return go back to where we were
5                        !
5                        CTX [ CTX$L_NEXT_VBN ] = .CTX [ CTX$L_SAVE_VBN ];
5
5                        ! Get the saved previous bucket
5                        !
5                        RECL$$GET_NEXT_BUCKET()
5
4                        END;
4
4                    ! Swap the suckers
4                    !
4                    RECL$$SWAP_BUFFERS();
4
4                    ! Get the saved bucket
4                    !
4                    RECL$$GET_NEXT_BUCKET();
4
4                    ! Return
4                    !
4                    EXITLOOP
4
4                    END
3                ELSE
4                    BEGIN
4
4                    ! Its not the last bucket, so go read the next bucket
4                    !
4                    RECL$$SWAP_BUFFERS();
4
4                    RECL$$GET_NEXT_BUCKET()
4
4                    END
4
3            END
3
2        UNTIL RECL$_FOREVER;
2
2        ! We exit the loop on sucess so return the context back to where it
2        ! was when we were called
2        !
2        CTX = .CTX - CTX$K_BLN;
2
2        BUCKET = .CTX [ CTX$L_CURRENT_BUFFER ];
2
2        RETURN .STATUS
2
1        END;
```

```
                                   010C    8F  BB 00000 UPDATE_INDEX:
                                                                  PUSHR    #^M<R2,R3,R8>                              : 0209
                           53              01  D0 00004           MOVL     #1, STATUS                                 : 0274
               03    0D    A9              01  E1 00007           BBC      #1, 13(BUCKET), 1$                         : 0278
                                         009F  31 0000C           BRW      14$                                        :
                           52      08      A9  D0 0000F 1$:       MOVL     8(BUCKET), NEXT_DATA_BUCKET                : 0285
                           5A      5C      AA  9E 00013           MOVAB    92(R10), CTX                               : 0289
                           59      04      AA  D0 00017           MOVL     4(CTX), BUCKET                             : 0293
                     54    AA      44      AA  D0 0001B           MOVL     68(CTX), 84(CTX)                           : 0297
                                   10      AE  DD 00020 2$:       PUSHL    VBN                                        : 0306
                                        0000G  30 00023           BSBW     RECL$$GET_DOWN_POINTER                     :
                           5E              04  C0 00026           ADDL2    #4, SP                                     :
                           56              50  E9 00029           BLBC     R0, 8$                                     :
                                        0000G  30 0002C           BSBW     RECL$$CHECK_LAST                           : 0328
                           04              50  E9 0002F           BLBC     R0, 3$                                     :
                                           53  D4 00032           CLRL     STATUS                                     : 0331
                                           70  11 00034           BRB      13$                                        : 0330
                           01      02      AA  91 00036 3$:       CMPB     2(CTX), #1                                 : 0335
                                           15  12 0003A           BNEQ     4$                                         :
                                           52  DD 0003C           PUSHL    NEXT_DATA_BUCKET                           : 0341
                                        0000G  30 0003E           BSBW     RECL$$COMPARE_POINTER                      :
                           5E              04  C0 00041           ADDL2    #4, SP                                     :
                           0A              50  E8 00044           BLBS     R0, 4$                                     :
                                           52  DD 00047           PUSHL    NEXT_DATA_BUCKET                           : 0353
                                        0000G  30 00049           BSBW     RECL$$SWING_POINTER                        :
                           5E              04  C0 0004C           ADDL2    #4, SP                                     :
                                           03  11 0004F           BRB      5$                                         : 0341
                                        0000G  30 00051 4$:       BSBW     RECL$$REMOVE_INDEX_RECORD                  : 0359
                                        0000G  30 00054 5$:       BSBW     RECL$$BUCKET_EMPTY                         : 0363
                           1D              50  E9 00057           BLBC     R0, 7$                                     :
                                   08      AA  DD 0005A           PUSHL    8(CTX)                                     : 0371
                                           A1  10 0005D           BSBB     UPDATE_INDEX                               :
                           5E              04  C0 0005F           ADDL2    #4, SP                                     :
                           53              50  D0 00062           MOVL     R0, STATUS                                 :
                           05              53  E9 00065           BLBC     STATUS, 6$                                 :
                                        0000V  30 00068           BSBW     REMOVE_BUCKET                              : 0377
                                           33  11 0006B           BRB      12$                                        : 0381
                     50    AA      54      AA  D0 0006D 6$:       MOVL     84(CTX), 80(CTX)                           : 0390
                                   08      AA  D4 00072           CLRL     8(CTX)                                     : 0394
                                           21  11 00075           BRB      10$                                        : 0398
                                   04      AA  9F 00077 7$:       PUSHAB   4(CTX)                                     : 0411
                                        0000G  30 0007A           BSBW     RECL$$WRITE_BUCKET                         :
                           5E              04  C0 0007D           ADDL2    #4, SP                                     :
                                           24  11 00080           BRB      13$                                        : 0406
                           17      0D      A9  E9 00082 8$:       BLBC     13(BUCKET), 11$                            : 0429
                     54    AA      08      AA  D1 00086           CMPL     8(CTX), 84(CTX)                            : 0436
                                           08  13 0008B           BEQL     9$                                         :
                     50    AA      54      AA  D0 0008D           MOVL     84(CTX), 80(CTX)                           : 0442
                                        0000G  30 00092           BSBW     RECL$$GET_NEXT_BUCKET                      : 0446
                                        0000V  30 00095 9$:       BSBW     RECL$$SWAP_BUFFERS                         : 0452
                                        0000G  30 00098 10$:      BSBW     RECL$$GET_NEXT_BUCKET                      : 0456
                                           09  11 0009B           BRB      13$                                        : 0431
                                        0000V  30 0009D 11$:      BSBW     RECL$$SWAP_BUFFERS                         : 0468
                                        0000G  30 000A0 12$:      BSBW     RECL$$GET_NEXT_BUCKET                      : 0470
                                         FF7A  31 000A3           BRW      2$                                         : 0302
                           5A      A4      AA  9E 000A6 13$:      MOVAB    -92(R10), CTX                              : 0481
                           59      04      AA  D0 000AA           MOVL     4(CTX), BUCKET                             : 0483
```

```
                                    50                 53  D0 000AE 14$:     MOVL    STATUS, R0                                   ; 0485
                                           010C        8F  BA 000B1          POPR    #^M<R2,R3,R8>                                ; 0487
                                                           05 000B5          RSB                                                  ;

; Routine Size:  182 bytes,    Routine Base:  _CONV$RECL_S + 0027


;  492          0488  1
```

```
1  %SBTTL  'REMOVE_BUCKET'
1  ROUTINE REMOVE_BUCKET : RL$JSB_REG_9 NOVALUE =
1  !++
1  !
1  ! Functional Description:
1  !
1  !       This routine takes the steps required to remove a bucket from the
1  !       horizontal chain, write it to the AVAIL list, and update the key
1  !       descriptor if necessary.
1  !
1  ! Calling Sequence:
1  !
1  !       REMOVE_BUCKET();
1  !
1  ! Input Parameters:
1  !       none
1  !
1  ! Implicit Inputs:
1  !
1  !       CTX to point to current bucket, etc...
1  !
1  ! Output Parameters:
1  !       none
1  !
1  ! Implicit Outputs:
1  !
1  !       The bucket is removed and written to the AVAIL list.  All pointers
1  !       are updated.
1  !
1  ! Routine Value:
1  !       none
1  !
1  ! Routines Called:
1  !
1  !       CONV$$WRITE_KEY_DESC
1  !       RECL$$WRITE_BUCKET
1  !       ZERO_BUCKET
1  !       CONV$$WRITE_AREA_DESC
1  !
1  ! Side Effects:
1  !
1  !       RECL$GL_DATA_COUNT is incremented if we reclaim a data bucket.
1  !       RECL$GL_INDEX_COUNT is incremented if we reclaim an index bucket.
1  !
1  !--
1
2      BEGIN
2
2      DEFINE_CTX;
2      DEFINE_BUCKET;
2      DEFINE_KEY_DESC;
2
2      EXTERNAL
2          RECL$GL_DATA_COUNT,
2          RECL$GL_INDEX_COUNT;
2
2      LOCAL
```

```
            AREA_DESC : REF BLOCK [ ,BYTE ];

        ! The removal of a bucket is done in three steps, the order of which
        ! is of the utmost importance to the reliability of the utility. It
        ! is assumed that the index record for this bucket has been removed.
        !
        ! Step I
        !
        ! Update the previous bucket pointer to point to the next one in the chain
        !
        BEGIN

        LOCAL PREVIOUS_BUCKET : REF BLOCK [ ,BYTE ];

        PREVIOUS_BUCKET = .CTX [ CTX$L_PREVIOUS_BUFFER ];

        ! Update the previous bucket in the chain
        !
        PREVIOUS_BUCKET [ BKT$L_NXTBKT ] = .CTX [ CTX$L_NEXT_VBN ];

        RECL$$WRITE_BUCKET( CTX [ CTX$L_PREVIOUS_BUFFER ] )

        END;

        ! Step Ia
        !
        ! In the case that this is the first bucket in a chain then either do
        ! nothing or update the key descriptor, depending on the level.
        !
        ! Is this the first bucket in the chain
        !
        IF .CTX [ CTX$L_CURRENT_VBN ] EQLU .CTX [ CTX$L_FIRST_VBN ]
        THEN
            BEGIN

            ! If this is the data level bucket then update the key descriptor
            ! else continue
            !
            IF .BUCKET [ BKT$B_LEVEL ] EQLU 0
            THEN
                BEGIN

                KEY_DESC [ KEY$L_LDVBN ] = .CTX [ CTX$L_NEXT_VBN ];

                CONV$$WRITE_KEY_DESC()

                END;

            ! The next vbn will now be the first in the chain
            !
            CTX [ CTX$L_FIRST_VBN ] = .CTX [ CTX$L_NEXT_VBN ]

            END;

        ! Step II
        !
        ! Update the current bucket to point to the first bucket in the area
        ! available list
        !
        ! To update the bucket we must use the area descriptor
        !
        AREA_DESC = .CONV$AR_AREA_BLOCK + ( .CTX [ CTX$B_AREA ] * AREA$K_BLN );

        ! Point the bucket to the first avail. bucket
        !
        BUCKET [ BKT$L_NXTBKT ] = .AREA_DESC [ AREA$L_AVAIL ];

        ! If first bucket on free list set the last bucket bit
        !
        IF .BUCKET [ BKT$L_NXTBKT ] EQLU 0
        THEN
            BUCKET [ BKT$V_LASTBKT ] = _SET;

        ! Zero the data portion of the bucket
        !
        ZERO_BUCKET();

        ! Write the bucket into the file
        !
        RECL$$WRITE_BUCKET( CTX [ CTX$L_CURRENT_BUFFER ] );

        ! Count the reclaimed bucket.
        !
        IF .BUCKET[ BKT$B_LEVEL ] EQLU 0
        THEN

            ! Its a data bucket we're reclaiming.
            !
            RECL$GL_DATA_COUNT = .RECL$GL_DATA_COUNT + 1
        ELSE

            ! Its an index bucket we're reclaiming.
            !
            RECL$GL_INDEX_COUNT = .RECL$GL_INDEX_COUNT + 1;


        ! Step III
        !
        ! Update the area descriptor with the new bucket at the head of the
        ! availiable list
        !
        AREA_DESC [ AREA$L_AVAIL ] = .CTX [ CTX$L_CURRENT_VBN ];

        CONV$$WRITE_AREA_DESC( .CTX [ CTX$B_AREA ] );

        RETURN

        END;


                                                .EXTRN  RECL$GL_DATA_COUNT
                                                .EXTRN  RECL$GL_INDEX_COUNT
```

```
                                                                              K 10
RECL$CTRL          VAX-11 CONVERT/RECLAIM                                    15-Sep-1984 23:58:52    VAX-11 Bliss-32 V4.0-742                       Page  17
V04-000            REMOVE_BUCKET                                             14-Sep-1984 12:14:03    [CONV.SRC]RECLCTRL.B32;1                             (6)

                                                         52  DD 00000 REMOVE_BUCKET:
                                                                             PUSHL     R2                                                    : 0490
                                  50         40          AA  D0 00002        MOVL      64(CTX), PREVIOUS_BUCKET                              : 0560
                            08    A0         50          AA  D0 00006        MOVL      80(CTX), 8(PREVIOUS_BUCKET)                           : 0564
                                             40          AA  9F 0000B        PUSHAB    64(CTX)                                               : 0566
                                                      0000G  30 0000E        BSBW      RECL$$WRITE_BUCKET                                    :
                                  5E                     04  C0 00011        ADDL2     #4, SP                                                :
                            24    AA         08          AA  D1 00014        CMPL      8(CTX), 36(CTX)                                       : 0577
                                                         12  12 00019        BNEQ      2$                                                    :
                                             0C          A9  95 0001B        TSTB      12(BUCKET)                                            : 0584
                                                         08  12 0001E        BNEQ      1$                                                    :
                            54    AB         50          AA  D0 00020        MOVL      80(CTX), 84(KEY_DESC)                                 : 0588
                                                      0000G  30 00025        BSBW      CONV$$WRITE_KEY_DESC                                  : 0590
                            24    AA         50          AA  D0 00028 1$:    MOVL      80(CTX), 36(CTX)                                      : 0596
                                  50         01          AA  9A 0002D 2$:    MOVZBL    1(CTX), R0                                            : 0607
                      50          50         06          78  00031           ASHL      #6, R0, R0                                            :
                      52          50      0000G          CF  C1 00035        ADDL3     CONV$AR_AREA_BLOCK, R0, AREA_DESC                     :
                            08    A9         08          A2  D0 0003B        MOVL      8(AREA_DESC), 8(BUCKET)                               : 0611
                                                         04  12 00040        BNEQ      3$                                                    : 0615
                            0D    A9                     01  88 00042        BISB2     #1, 13(BUCKET)                                        : 0617
                                                      0000V  30 00046 3$:    BSBW      ZERO_BUCKET                                           : 0621
                                             04          AA  9F 00049        PUSHAB    4(CTX)                                                : 0625
                                                      0000G  30 0004C        BSBW      RECL$$WRITE_BUCKET                                    :
                                  5E                     04  C0 0004F        ADDL2     #4, SP                                                :
                                             0C          A9  95 00052        TSTB      12(BUCKET)                                            : 0629
                                                         06  12 00055        BNEQ      4$                                                    :
                                          0000G          CF  D6 00057        INCL      RECL$GL_DATA_COUNT                                    : 0634
                                                         04  11 0005B        BRB       5$                                                    :
                                          0000G          CF  D6 0005D 4$:    INCL      RECL$GL_INDEX_COUNT                                   : 0639
                            08    A2         08          AA  D0 00061 5$:    MOVL      8(CTX), 8(AREA_DESC)                                  : 0647
                                  51         01          AA  9A 00066        MOVZBL    1(CTX), R1                                            : 0649
                                                      0000G  30 0006A        BSBW      CONV$$WRITE_AREA_DESC                                 :
                                                         04  BA 0006D        POPR      #^M<R2>                                               : 0653
                                                             05 0006F        RSB                                                             :

; Routine Size:  112 bytes,    Routine Base:  _CONV$RECL_S + 00DD


;  659          0654  1
```

```
0655 1  %SBTTL  'ZERO_BUCKET'
0656 1  ROUTINE ZERO_BUCKET : RL$JSB_REG_9 NOVALUE =
0657 1  !++
0658 1  !
0659 1  ! Functional Description:
0660 1  !
0661 1  !       Zeros out the data portion of a index bucket
0662 1  !
0663 1  ! Calling Sequence:
0664 1  !
0665 1  !       ZERO_BUCKET()
0666 1  !
0667 1  ! Input Parameters:
0668 1  !       none
0669 1  !
0670 1  ! Implicit Inputs:
0671 1  !       none
0672 1  !
0673 1  ! Output Parameters:
0674 1  !       none
0675 1  !
0676 1  ! Implicit Outputs:
0677 1  !       none
0678 1  !
0679 1  ! Routine Value:
0680 1  !       none
0681 1  !
0682 1  ! Routines Called:
0683 1  !       none
0684 1  !
0685 1  ! Side Effects:
0686 1  !       none
0687 1  !
0688 1  !--
0689 1
0690 2     BEGIN
0691 2
0692 2     DEFINE_CTX;
0693 2     DEFINE_BUCKET;
0694 2     DEFINE_KEY_DESC;
0695 2
0696 2     CH$FILL( 0,                                                  ! Fill with 0's
0697 2            .CTX [ CTX$W_BUCKET_SIZE ] - BKT$K_OVERHDSZ - 1,      ! This much
0698 2            .CTX [ CTX$L_CURRENT_BUFFER ] + BKT$K_OVERHDSZ );     ! Starting here
0699 2
0700 2     RETURN
0701 2
0702 1     END;



                        3C  BB 00000 ZERO_BUCKET:
                                         PUSHR   #^M<R2,R3,R4,R5>                     ; 0656
      51        58  AA  3C 00002         MOVZWL  88(CTX), R1                          ; 0697
      51            0F  C2 00006         SUBL2   #15, R1                              ;
```

```
                                   50      04   AA  D0 00009        MOVL    4(CTX), R0                                    ; 0698
          51          00           6E           00  2C 0000D        MOVC5   #0, (SP), #0, R1, 14(R0)                      ;
                                           0E   A0     00012                                                              ;
                                                3C  BA 00014        POPR    #^M<R2,R3,R4,R5>                              ; 0702
                                                    05 00016        RSB                                                   ;

; Routine Size:  23 bytes,    Routine Base:  _CONV$RECL_S + 014D


;  709          0703  1
```

```
1 %SBTTL  'SWAP_BUFFERS'
1 GLOBAL ROUTINE  RECL$$SWAP_BUFFERS : RL$JSB_REG_9 NOVALUE =
1 !++
1 !
1 ! Functional Description:
1 !
1 ! Calling Sequence:
1 !
1 ! Input Parameters:
1 !       none
1 !
1 ! Implicit Inputs:
1 !       none
1 !
1 ! Output Parameters:
1 !       none
1 !
1 ! Implicit Outputs:
1 !       none
1 !
1 ! Routine Value:
1 !       none
1 !
1 ! Routines Called:
1 !       none
1 !
1 ! Side Effects:
1 !       none
1 !
1 !--
1
2     BEGIN
2
2     DEFINE_CTX;
2     DEFINE_BUCKET;
2     DEFINE_KEY_DESC;
2
2     LOCAL
2         TEMP_BUF,
2         TEMP_VBN;
2
2     ! Swap the current buffer with the previous buffer and change bucket
2     !
2     TEMP_BUF = .CTX [ CTX$L_PREVIOUS_BUFFER ];
2     TEMP_VBN = .CTX [ CTX$L_PREVIOUS_VBN ];
2
2     CTX [ CTX$L_PREVIOUS_BUFFER ] = .CTX [ CTX$L_CURRENT_BUFFER ];
2     CTX [ CTX$L_PREVIOUS_VBN ]    = .CTX [ CTX$L_CURRENT_VBN ];
2
2     CTX [ CTX$L_CURRENT_BUFFER ] = .TEMP_BUF;
2     CTX [ CTX$L_CURRENT_VBN ]    = .TEMP_VBN;
2
2     BUCKET = .TEMP_BUF;
2
2     RETURN
2
1     END;
```

```
                                    50       40   AA  7D 00000 RECL$$SWAP_BUFFERS::
                                                                        MOVQ    64(CTX), TEMP_BUF                          ; 0747
                              40    AA       04   AA  7D 00004          MOVQ    4(CTX), 64(CTX)                            ; 0750
                              04    AA            50  7D 00009          MOVQ    TEMP_BUF, 4(CTX)                           ; 0753
                                    59            50  D0 0000D          MOVL    TEMP_BUF, BUCKET                           ; 0756
                                                      05 00010          RSB                                                ; 0760

; Routine Size:  17 bytes,    Routine Base:  _CONV$RECL_S + 0164


;  768          0761  1
;  769          0762  0 END      ELUDOM




;                                   PSECT SUMMARY
;
;       Name                        Bytes                        Attributes
;
;  _CONV$RECL_S                         373  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)




;                           Library Statistics
;
;                                                 -------- Symbols --------        Pages        Processing
;       File                                      Total    Loaded   Percent       Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1                  18619         8         0        1000          00:01.8
;  _$255$DUA28:[CONV.SRC]CONVERT.L32;1              165        23        13          17          00:00.2




;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RECLCTRL/OBJ=OBJ$:RECLCTRL MSRC$:RECLCTRL/UPDATE=(ENH$:RECLCTRL)

; Size:          373 code + 0 data bytes
; Run Time:         00:12.9
; Elapsed Time:     00:36.2
; Lines/CPU Min:      3538
; Lexemes/CPU-Min: 13500
; Memory Used:  103 pages
; Compilation Complete
```

0066 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY